Agile Principles and Mindset

DOMAIN I

What is Agile

- Developed for Software projects, but it is a methodology that can be used on all Projects types
- Agile is an umbrella term that is used to refer to different types of iterative development
- Scrum is the most common method of agile, there are others such as extreme programming (XP), lean development, and Kanban.

Agile vs. Traditional Project Management

- Agile builds in increments vs. as a whole
- Agile does planning throughout vs. done all at once
- Agile delivers products over time vs. all at once
- Customers sees value faster vs. at the end
- Agile wants changes vs. discouraging changes

Agile Benefits

- Customer involved throughout the life cycle
- Greater Customer Interaction with all stakeholders
- Constant Feedback is required to stay current and successful
- Greater Value up front
- Change is welcomed by all stakeholders

Agile Concurrent Development

- Fund incrementally – opt to extend, redirect or cancel at a very granular level
- Deliver & realize value steadily
- Validate designs with users & customers
- Continuously adapt to risk and change
- Integrate early & often

Agile *Declaration of Interdependence (DOI)*

Agile and adaptive approaches for linking people, projects and value

We are a community of project leaders that are highly successful at delivering results. To achieve these results:

*We **increase return on investment** by making continuous flow of value our focus.*

*We **deliver reliable results** by engaging customers in frequent interactions and shared ownership.*

*We **expect uncertainty** and manage for it through iterations, anticipation, and adaptation.*

*We **unleash creativity and innovation** by recognizing that individuals are the ultimate source of value, and creating an environment where they can make a difference.*

*We **boost performance** through group accountability for results and shared responsibility for team effectiveness.*

*We **improve effectiveness and reliability** through situationally specific strategies, processes and practices.*

Agile Mindset

Welcoming change

Working in small value increments

Using build and feedback loops

Learning through discovery

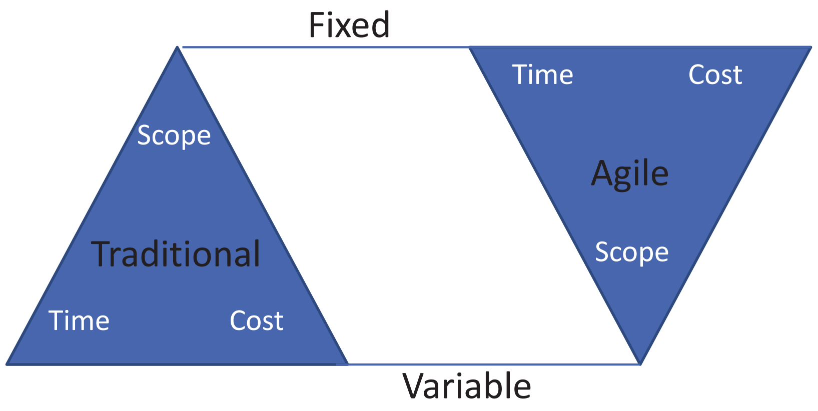
Value -driven development

Failing fast with learning

Continuous delivery

Continuous improvement

Inverting the Triangle



Agile Manifesto

Create in 2001

Contains:

- 4 values
- 12 guiding principles

https://agilemanifesto.org/

The Agile Manifesto Values

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals & interactions	over	Processes & tools
Working software	over	Comprehensive documentation
Customer collaboration	over	Contract negotiation
Responding to change	over	Following a plan

That is, while there is value in the items on the **right**,
we value the items on the **left** more.

www.agilemanifesto.org

Individuals and interactions over processes and tools

- While processes and tools will likely be necessary on our projects, we should focus the team's attention on the individuals and interactions involved.
- Projects are undertaken by people, not tools
- Problems get solved by people, not processes
- Projects are ultimately about people

Working software over comprehensive documentation

- Focus on the delivering value vs. paperwork.
- Agile documents should be barely sufficient
- Done just in time
- Done just because
- Delivering software that does what it should comes first, before creating documentation.
- Agile dramatically simplify the administrative paperwork relating to time, cost, and scope control

Customer collaboration over contract negotiation

- Be flexible and accommodating, instead of fixed and uncooperative
- Manage change, don’t suppress change
- Shared definition of “done”
- Requires trusting relationship

Responding to change over following a plan

- Spend effort and energy responding to changes
- Software projects tend to have high rates of change

Agile Guiding Principles 1-3

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Agile Guiding Principles 4-6

4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Agile Guiding Principles 7-9

7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers)and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.

Agile Guiding Principles 10-12

10. Simplicity; the art of maximizing the amount of work not done is essential.
11. The best architectures, requirements and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective then tunes and adjusts its behavior accordingly.

Agile Methods

- Over 12 agile methodologies
- Scrum
- Extreme Programming (XP)
- Kanban Development
- Lean Software Development

Agile Terms

Product Owner - Designated person that represents the customer on the project

Agile Project Manager/Scrum Master – Manages the agile project

Product Backlog - Project requirements from the stakeholders

Sprint Planning Meeting- Meeting done by the agile team to determine what features will be done in the next sprint

Sprint Backlog – Work the team selects to get done in the next sprint

Sprint - A short iteration where the project teams work to complete the work in the sprint backlog, (1-4 weeks typical)

Daily Stand Up Meeting - A quick meeting each day to discuss project statuses, led by the Scrum Master. Usually 15 minutes

Sprint Review – An inspection done at the end of the sprint by the customers

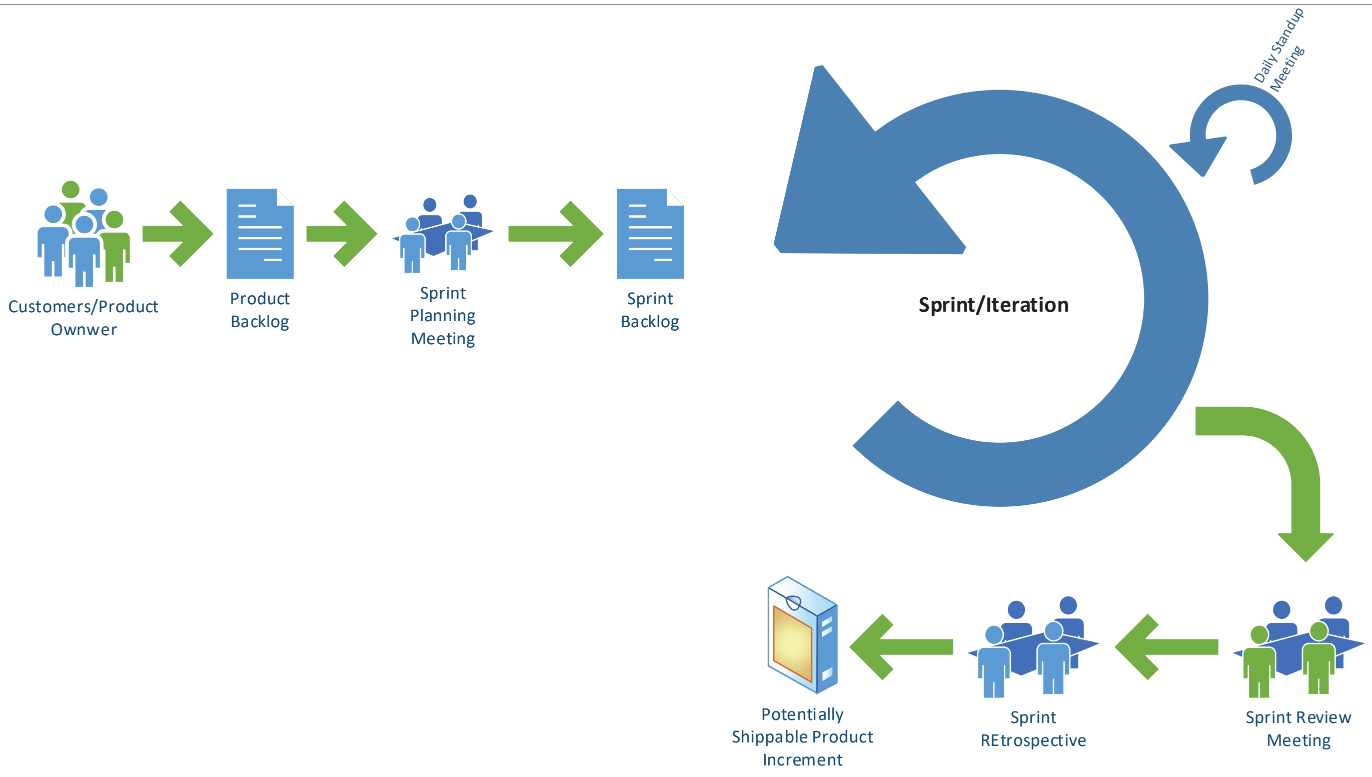
Retrospective – Meeting done to determine what went wrong during the sprint and what when right. Lesson learned for the sprint.

Partial Completed Product - Customers Demo the product and provides feedback. This feedback adjust the next Sprint priorities

Release - Several Sprints worth of work directed to operations for possible rollout and testing

Sprint = Iteration

Agile Process



Scrum

Set of team guidance practices, roles, events, artifacts, and rules

Based on three pillars of Transparency, Inspection, and Adaptation:

- Transparency
  - Visibility to those responsible for the outcome
- Inspection
  - Timely checks on how well a project is progressing toward goals
  - Looks for problematic deviations or differences from goals
- Adaptation
  - Adjusting a process to minimize further issues if an inspection shows a problem or undesirable trend

Scrum Roles & Responsibilities

Product Owner

- Owns Product vision
- Defines features, decides on release date and content
- Responsible for market success
- Prioritizes features according to market value
- Can change features and priorities every Sprint

ScrumMaster

- Responsible for facilitating process
- Focuses Team and protects them from external interruption
- Looks for ways to enhance productivity
- Assists Product Owner in leveraging Scrum

Scrum Roles & Responsibilities

Development Team

- Small group containing all necessary project skills
- Focuses on steady delivery of high quality features
- Generates options for delivery
- Manages own work within Sprints

Scrum Activities

The Scrum methodology refers to several different types of activities:

1. sprint planning meeting
2. sprints
   - Daily stand-up meeting
3. sprint review meeting
4. sprint retrospectives.

Sprint Planning Meeting

- ✓Used to determine what work will be done in that sprint and how the work will be achieved.
- ✓The development team predicts what can be delivered based on estimates, projected capacity, and past performance to define the sprint goal.
- ✓The development team then determines how this functionality will be built and how the team will organize to deliver the sprint goal.
- ✓Output of this will be the sprint backlog. The work to get done in the next sprint.

Sprints

- ✓A sprint is a **timeboxed** (time-limited) iteration of 1-4 weeks to build a potentially releasable product
- ✓Each sprint includes a sprint planning meeting, daily Scrum, the actual work, a sprint review meeting, and the sprint retrospective
- ✓During the sprint, no changes are made that would affect the sprint
- ✓The development team members are kept the same throughout the sprint

Daily Scrum (or Standup)

- ✓A 15-minute time-boxed activity for the Development Team to synchronize activities and create a plan for the next 24 hours
- ✓Should be held at the same time and place each day
- ✓Each team member should answer 3 questions:
  1. What did you do yesterday?
  2. What will you do today?
  3. Are there any impediments in your way?

Sprint Review

- ✓Takes place at the end of the Sprint
- ✓Designed to gather feedback from stakeholders on what the Team has completed in the sprint
- ✓Team demonstrates work that was completed during the sprint
- ✓To create a conversation between the Team and the stakeholders about how to make the product better
- ✓should be time boxed to no more than an hour per week of Sprint

Sprint Retrospective

- ✓Opportunity for the Team to inspect and create a plan for improvements to be done during the next Sprint.
- ✓Team discusses:
  - ✓What went well
  - ✓What went wrong
  - ✓What to do more of
  - ✓What to do less of

Scrum Artifacts

- ✓ Product increment
  - ✓ Part of the product that is complete after each sprint
- ✓ Product Backlog
  - ✓ Prioritized list of valuable items to deliver
- ✓ Sprint Backlog
  - ✓ List of committed items to be addressed within Sprint

Product Backlog

- ✓Prioritized list of all work that needs to be done to complete the product
- ✓List is dynamics, it evolves as the more work is added and prioritized
- ✓Items in it is prioritized by the product owner and is sorted by value
- ✓Most valuable items are listed first
- ✓Constantly being refined as more work is added to it.
- ✓Team and product owner will “groom the backlog”.

Product Increment

- ✓Part of the product that is done after each sprint
- ✓Done to get feedback after each sprint
- ✓The product owner and team needs to agree upon the “definition of done” before the team starts working on the product

Sprint Backlog

- ✓The sprint backlog is the set of items from the product backlog that were selected for a specific sprint.
- ✓The sprint backlog is accompanied by a plan of how to achieve the sprint goal, so it serves as the development team's forecast for the functionality that will be part of the sprint.
- ✓It is a highly visible view of the work being undertaken and may only be updated by the development team.

Definition of Done (DoD)

Definition of Done (DoD) is a shared understanding of what it means when work is considered done, it should be defined at the beginning of the project, and it applies globally to the project.

Definition of Done (DoD) is a crucial element of a successful scrum software development

Might include things such as:

- DoD for Unit & functional tests.
- DoD Documentation.
- DoD for a Writing code.

Extreme Programming (XP)

Software development centric agile method

Focus software development good practices

Scrum at the project management level focuses on prioritizing work and getting feedback

XP Core Values

Simplicity

- Reduce complexity, extra features, and waste
- “ Find the simplest thing that could possibly work"

Communication

- Team members know what is expected of them and what other people are working on
- Daily stand-up meeting is key communication component

Feedback

- Get impressions of correctness early
- Failing fast allows for faster improvement

XP Core Values

Courage
- Allow our work to be entirely visible to others

Respect
- People work together as a team and everyone is accountable for the success or failure of the project
- Recognize people work differently and respect those differences

XP Roles

Coach

- Acts as a mentor, guiding the process and helping the team stay on track. Is a facilitator helping the team become effective.

Customer:

- Business representative who provides the requirements, priorities, and drives the business direction for the project.

Programmers

- Developers who build the product. Writes the codes.

Testers

- Helps the customer define and write the acceptance tests for the user stories.

Product Owner and Customer are equivalent
ScrumMaster and Coach are equivalent

XP Practices

Planning Activities (Games):

- Release Planning:
  - Push of new functionality all the way to the production user
  - Customer outlines the functionality required
  - Developers estimate difficult build
- Iteration Planning:
  - Short development cycles within a release (Scrum calls "sprints")
  - Conducted at start of every iteration, or every two weeks
  - Customer explains functionality they would like in iteration
  - Developers break functionality into tasks and estimate work
  - Based on estimates and amount of work accomplished in previous iteration,

XP Practices

Small Releases:

- Frequent, small releases to test environments
- Demonstrate progress and increase visibility for the customer
- Quality is maintained: Rigorous testing or Continuous integration

Customer Tests:

- Customer describes one or more tests to show software is working
- Team builds automated tests to prove software is working.

Collective Code Ownership:

- Any pair of developers can improve or amend any code
- Multiple people work on all code, which results in increased visibility and knowledge of code base
- Leads to a higher level of quality; with more people looking at the code, there is a greater chance defects will be discovered.
- Less risk if programmer leaves, since knowledge is shared

XP Practices

Code Standards:

- Follow consistent coding standard
- Code looks as if it has been written by a single, knowledgeable programmer

Sustainable Pace:

- While periods of overtime might be necessary, repeated long hours of work are unsustainable and counterproductive
- The practice of maintaining a sustainable pace of development optimizes the delivery of long-term value

XP Practices

Metaphor:

- XP uses metaphors and similes to explain designs and create a shared technical vision.
- These descriptions establish comparisons that all the stakeholders can understand to help explain how the system should work.
- For example, “The invoicing module is like an Accounts receivable personnel who makes sure money collected from our customers”.

Continuous Integration:

- Integration involves bringing the code together and making sure it all compiles and works together.
- This practice is critical, because it brings problems to the surface before more code is built on top of faulty or incompatible designs.

XP Practices

Test -Driven Development (TDD):

- The team writes tests prior to developing the new code.
- If the tests are working correctly, the initial code that is entered will fail the tests
- The code will pass the test once it is written correctly.

Pair Programming:

- In XP, production code is written by two developers working as a pair to write and provide real-time reviews of the software as it emerges.
- Working in pairs also helps spread knowledge about the system through the team.

XP Practices

Simple Design:

- Code is always testable, browsable, understandable, and explainable
- Do the simplest thing that could possibly work next. Complex design is replaced with simpler design
- The best architectures, requirements, and designs emerge from self-organizing teams

Refactoring:

- Remove redundancy, eliminate unused functionality, and rejuvenate obsolete designs
- Refactoring throughout the entire project life cycle saves time and increases quality
- Code is kept clean and concise so it is easier to understand, modify, and extend

Some Basic Terminology Review

Scrum	Extreme Programming (XP)	Definition
Sprint	Iteration	Fixed-length period of time (timebox)
Release	Small Release	Release to production
Sprint/Release Planning	Planning Game	Agile planning meetings
Product Owner	Customer	Business representative to project
Retrospective	Reflection	“Lessons learned”-style meeting
ScrumMaster	Coach	Agile project manager
Development Team	Team	Empowered Cross-Functional team
Daily Scrum	Daily Standup	Brief daily status meeting

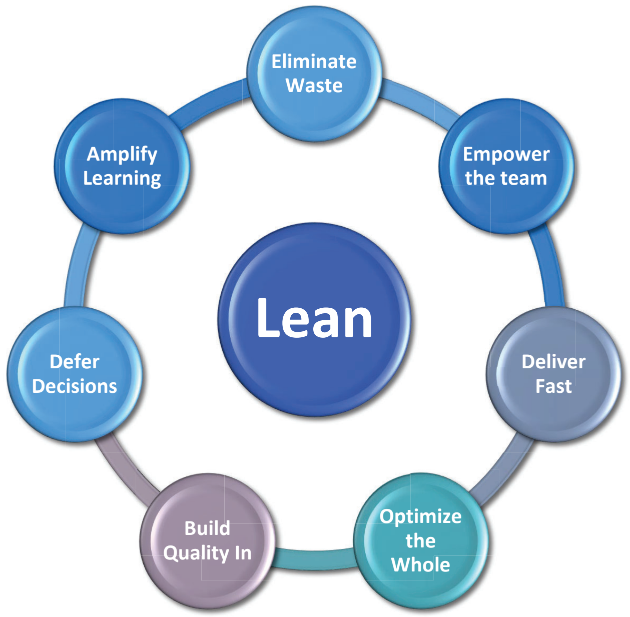
Lean Software Development

Lean was started by Toyota as manufacturing method that was applied to software development.

Principles:

- Using visual management tools
- Identifying customer-defined value
- Building in learning and continuous improvement

Lean Software Development



Lean Software Development

Eliminate waste:

- To maximize value, we must minimize waste. For software systems, waste can take the form of **partially done work**, **delays**, **handoffs, unnecessary features.**

Empower the team:

- Rather than taking a micro-management approach, we should respect team member's superior knowledge of the technical steps required on the project and let them

Deliver fast:

- Quickly delivering valuable software and iterating through designs.

Optimize the whole:

- We aim to see the system as more than the sum of its parts.

Lean Software Development

Build quality in:

- Build quality into the product and continually assure quality throughout the development process

Defer decisions:

- Balance early planning with making decisions and committing to things as late as possible.

Amplify learning:

- This concept involves facilitating communication early and often, getting feedback as soon as possible, and building on what we learn.

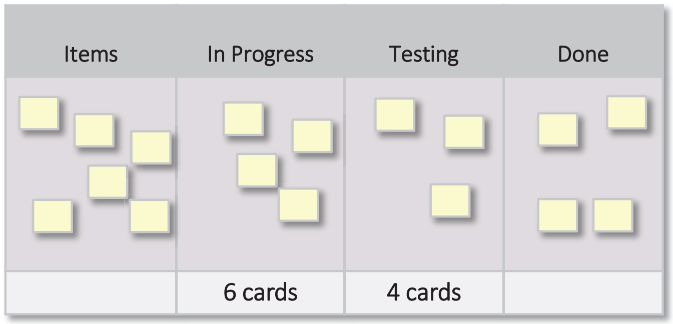
Seven Wastes of Lean

1. Partially done work
2. Extra Processes
3. Extra features
4. Task switching
5. Waiting
6. Motion
7. Defects

Kanban Development

Kanban development is derived from the lean production system used at Toyota.

"**Kanban**" is a Japanese word meaning "**signboard**."



Kanban five core principles:

Visualize the workflow:

- Software projects, by definition, manipulate knowledge, which is intangible and invisible.

Limit WIP:

- Keeping the amount of work in progress low increases the visibility of issues and bottlenecks

Manage flow:

- By tracking the flow of work through a system, issues can be identified and changes can be measured for effectiveness

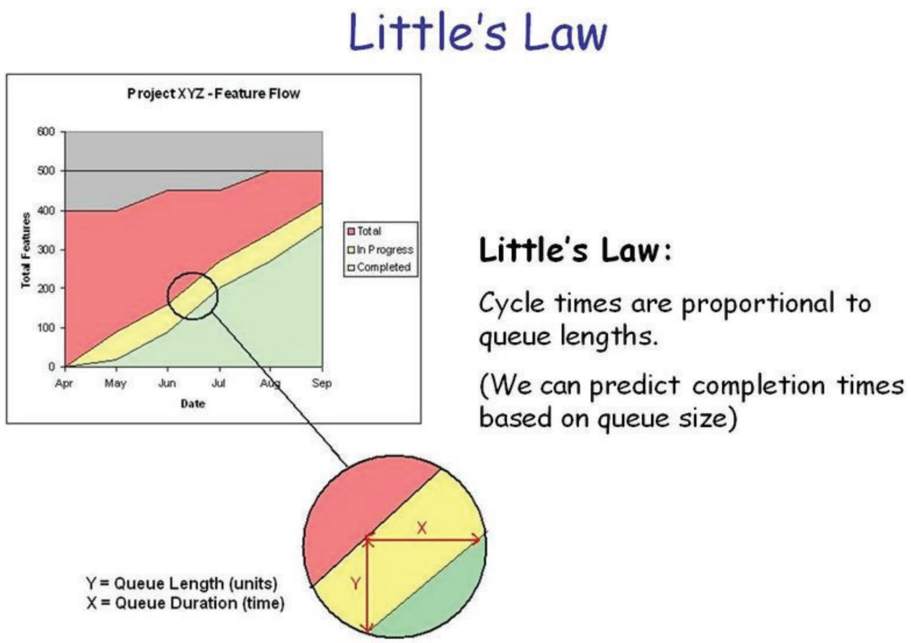
Make process policies explicit:

- It is important to clearly explain how things work so the team can have open discussions about improvements

Improve collaboration:

- Through scientific measurement and experimentation, the team should collectively own and improve the processes it uses.

Kanban Limit Work in Progress



Little's Law

Little's Law:

Cycle times are proportional to queue lengths.

(We can predict completion times based on queue size)

https://herdingcats.typepad.com/my_weblog/2014/08/the-use-and-misuse-of-littles-law.html

Other agile methods

Feature-Driven Development

- Team will first develop an overall model for the product then build a list, and plan the work.

Dynamic Systems Development

- One of the first agile methods and follows eight principles.

Crystal

- It’s a customized methodologies that are coded by color names.

Leading Effectively

Tap into people's intrinsic motivations

- Discover why team members want to do something and what motivates and then align that to the project goals

Management vs Leadership

- Management → Mechanical Focus
- Leadership → Humanistic Focus (on people and purpose)

Management Focus	Leadership Focus
Task/things	People
Control	Empowerment
Command	Communication

Leading Effectively

Servant Leadership

- Leader provides what the team needs
  1. Shield team from interruptions
  2. Remove impediments to progress
  3. (Re)Communicate project vision
  4. Carry food and water

Twelve Principles for Leading Agile Projects

1. Learn the team members needs
2. Learn the project requirements
3. Act for the simultaneous welfare of the team and the project
4. Create an environment of functional accountability
5. Have a vision of the completed project
6. Use the project vision to drive your own behavior

©2002 Jeffery Pinto, Project Leadership from Theory to Practice

Twelve Principles for Leading Agile Projects

7. Serve as the central figure in successful project team development
8. Recognize team conflict as a positive step
9. Manage with an eye toward ethics
10. Remember that ethics is not an afterthought, but an integral part of our thinking
11. Take time to reflect on the project
12. Develop the trick of thinking backwards

Leadership Tools and Techniques

Using these tools still need soft-skills approach

Modeling Desired Behavior
- Honesty
- Forward-Looking
- Competent
- Inspiring

Communicating project vision

Enabling others to act
- Switch from exclusive tools to inclusive tools

Being willing to change the status quo

Leadership Task

Practice Transparency through Visualization

Crate a safe environment for experimentation

Experiment with new techniques and processes

Share knowledge through collaboration

Encourage emergent leadership vis a safe environment

Value-Driven Delivery

Value-Driven Delivery

Projects undertaken to generate business value

- Produce Benefit
- Improve Service
- Market Demand
- Safety Compliance
- Regulatory Compliance

Early Value Delivery

Agile promote early and often delivery

Aim to deliver highest value early in project

- Deliver as many high-value components as soon as possible
  - Reduces risk
- Stakeholder satisfaction ➔ Project success
  - Shows understanding of stakeholders' needs
  - Stakeholders are engaged
  - Builds confidence of stakeholders in team

Reduce Waste

Minimize Waste, E.g:

- Partially done work
- Extra processes
- Extra features
- Waiting
- Defects

Assessing Value - Financial Metrics

Return on investment (ROI)

- The ratio of the benefits received from an investment to the money invested. Usually a percentage

Internal rate of return (IRR)

- Interest rate you will need to get in today's money to receive a certain amount of money in the future

Present Value/Net Present value (NPV)

- Value of future money in today's terms

Assessing Value - Financial Metrics

Earned Value Management

- Formulas that monitor the value of the project as its progressing.

Accounting on Agile Projects

Refers to how the different economic models of agile works

Agile accounting is different than traditional accounting

Agile looks to deliver value as quickly as possible

Uses minimal viable product (MVP)

This leads to more opportunity for incremental funding

Key Performance Indicators (KPI’s)

Uses as a way to measure the project progress

- Rate of progress: How much points has been completed
- Remaining work: How much work is yet to be done from the backlog
- Likely completion date
- Likely Cost remaining

Regulatory Compliance

Mandated requirements usually by government agencies

Must be implement into the project work as regular development work

Doing it after the project work is done

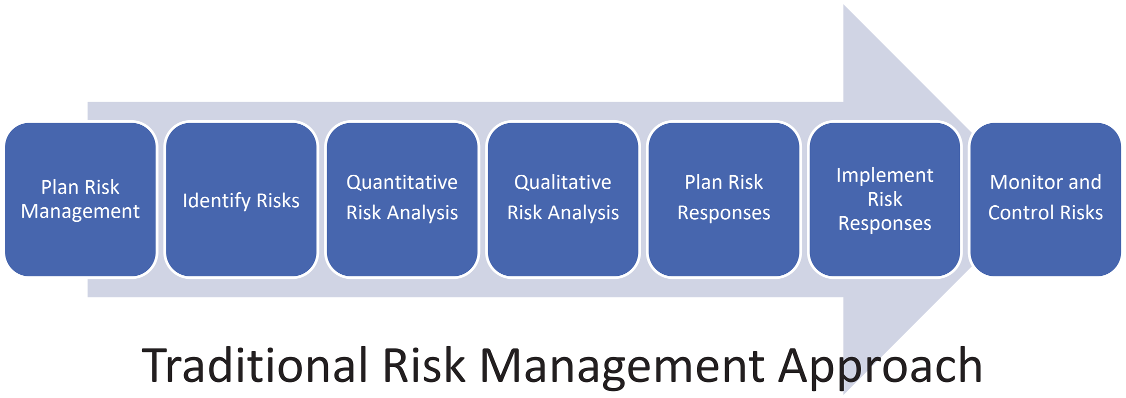
Risk Management

Risk is closely related to value

Considered as anti-value

Usually has the potential to remove, erode or reduce value with threats

Managing Risks Process



Traditional Risk Management Approach

Tools to Manage Risk

Risk-adjusted backlog

Risk burndown chart

How Customers Conduct Value Prioritization

Valued based prioritization is the one of core practices in agile planning

Features are prioritized on the basis of business value, risk and dependencies

Some of prioritization techniques used:

- Simple Scheme
- MoSCoW prioritization
- Monopoly Money
- 100-point method
- Dot Voting or Multi-voting
- Kano Analysis
- Requirements Prioritization Model

Prioritization Techniques

Simple Scheme

- Priority 1, Priority 2, Priority 3, etc.
- Could be problematic as many items might become the first priority.

MoSCoW prioritization

- Must have
- Should have
- Could have
- Would like to have, but not this time

Prioritization Techniques

Dot Voting or Multi-voting

- Each person gets a certain number of dots to distribute to the requirements

Monopoly Money

- Give everyone equal monopoly money
- They then distribute the funds to what they value the most

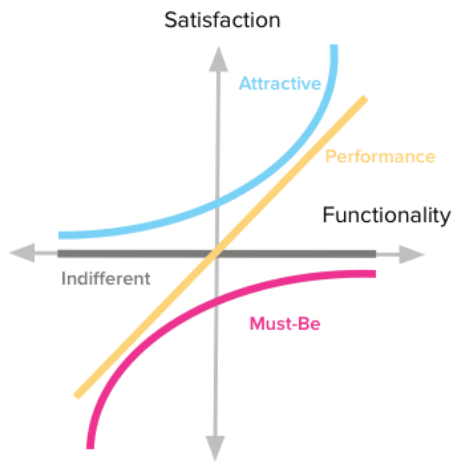
100-point method

- Each person is given 100 points
- They then use that to distribute to individual requirements

Prioritization Techniques

Kano Analysis

- Helps to understand the customers satisfaction
  - Delighters/Exciters
  - Satisfiers
  - Dissatisfiers
  - Indifferent



https://foldingburritos.com/kano-model/

Prioritization / Ranking is Relative

Doesn't matter what techniques the customers uses priority, the end results should be a list of prioritized features.

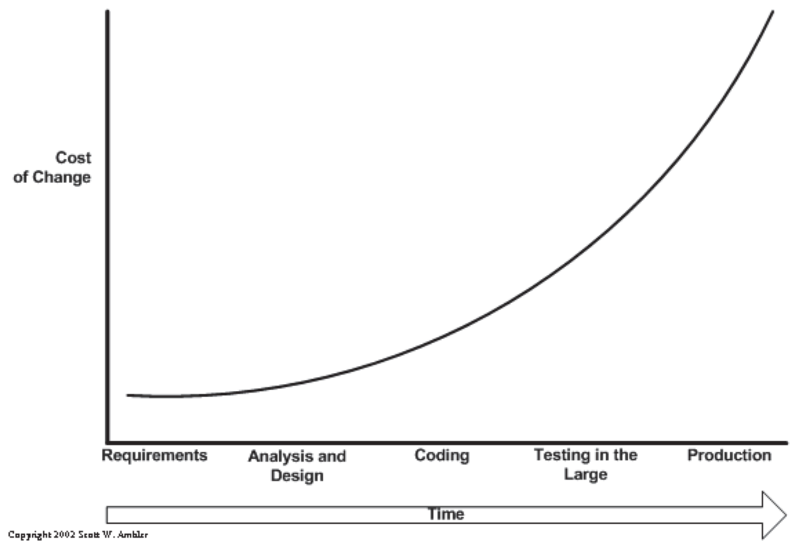
Delivering Value Incrementally

Incremental delivery is about deploying working parts of a product over the life of the project

In software development, its first delivered to a testing environment then to production

This will reduce the amount of rework by discovering issues early and fixing them

Delivering Value Incrementally



http://www.agilemodeling.com/essays/costOfChange.htm

Minimal Viable Product (MVP)

Refers to a set of functionality that is complete to be useful, but small enough not to be an entire project

Usually a module in a software

Tools for Agile Projects

Low-tech, high-touch over computer models

When using computer models problems could arise such as:

- Data accuracy perception increases
- No stakeholder interaction. Only a few people would update them

Low-Tech, High-Touch Tools

Use card, charts, whiteboards, and walls

Promotes communication and collaboration

Skip using a computer Gantt chart to a Kanban board

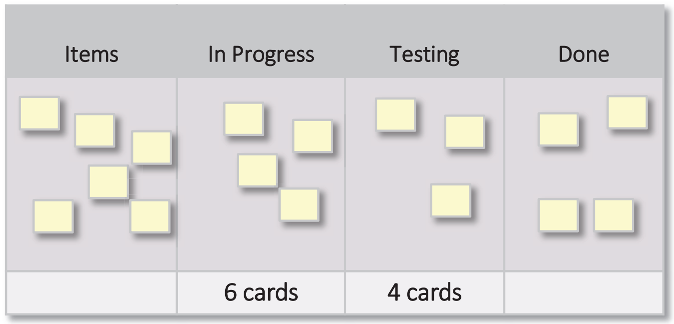
Kanban/Task Board

An "information radiator" - ensures efficient diffusion of information

Can be drawn on a whiteboard or even a section of wall

Makes iteration backlog visible

Serves as a focal point for the daily meeting



Limit WIP (Work in Progress)

Includes work that has been started but not completed yet

Represents money spent with no return

Hides process bottlenecks that slow the processes

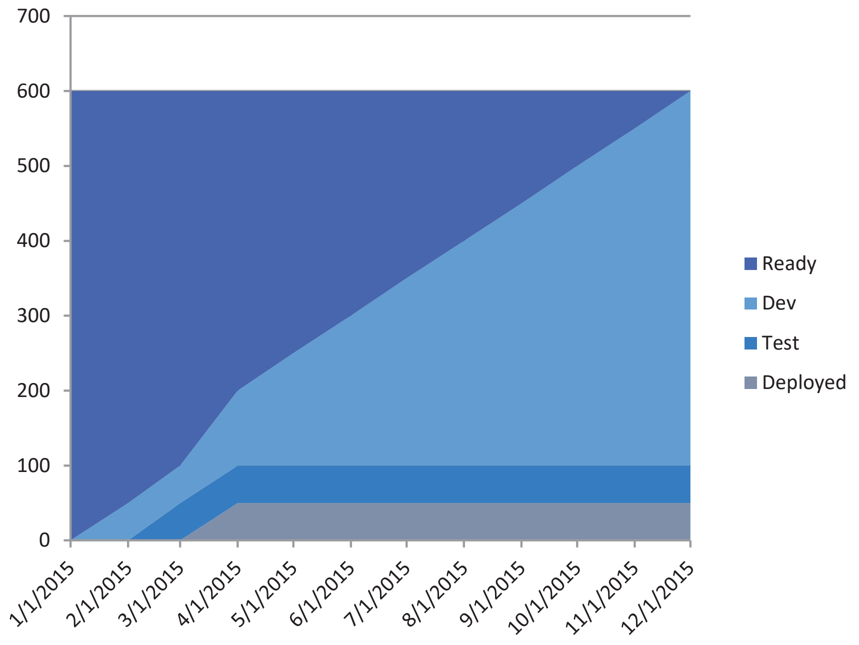
Represents risk in form of potential risk

Agile processes aim to Limit and optimize WIP

Optimal WIP makes processes efficient

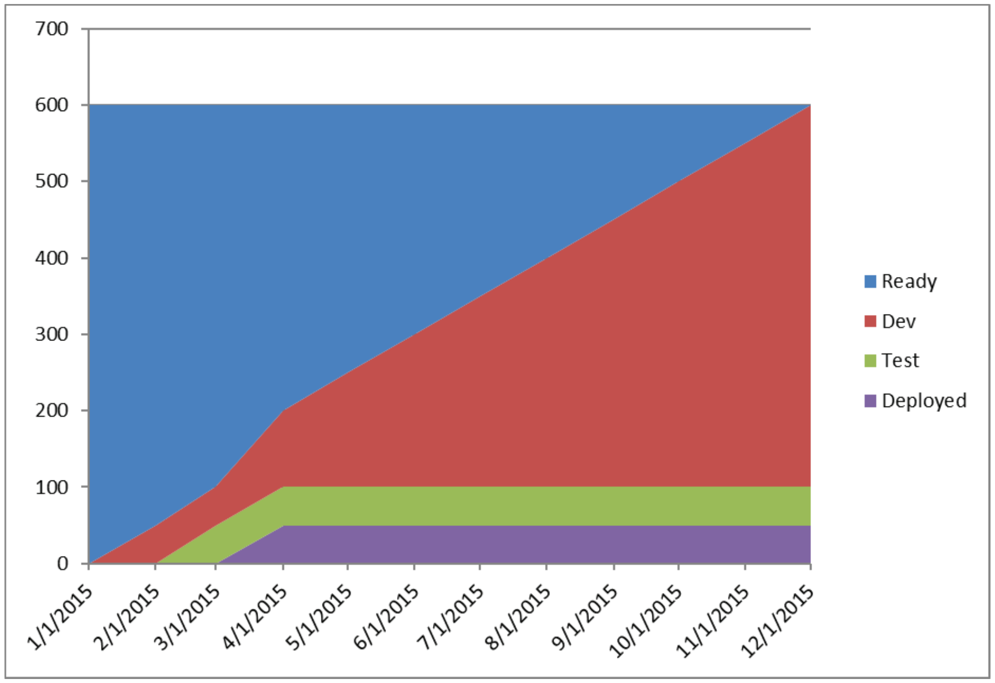
Cumulative Flow Diagrams (CFD's)

Stack graphs that show how work is progressing



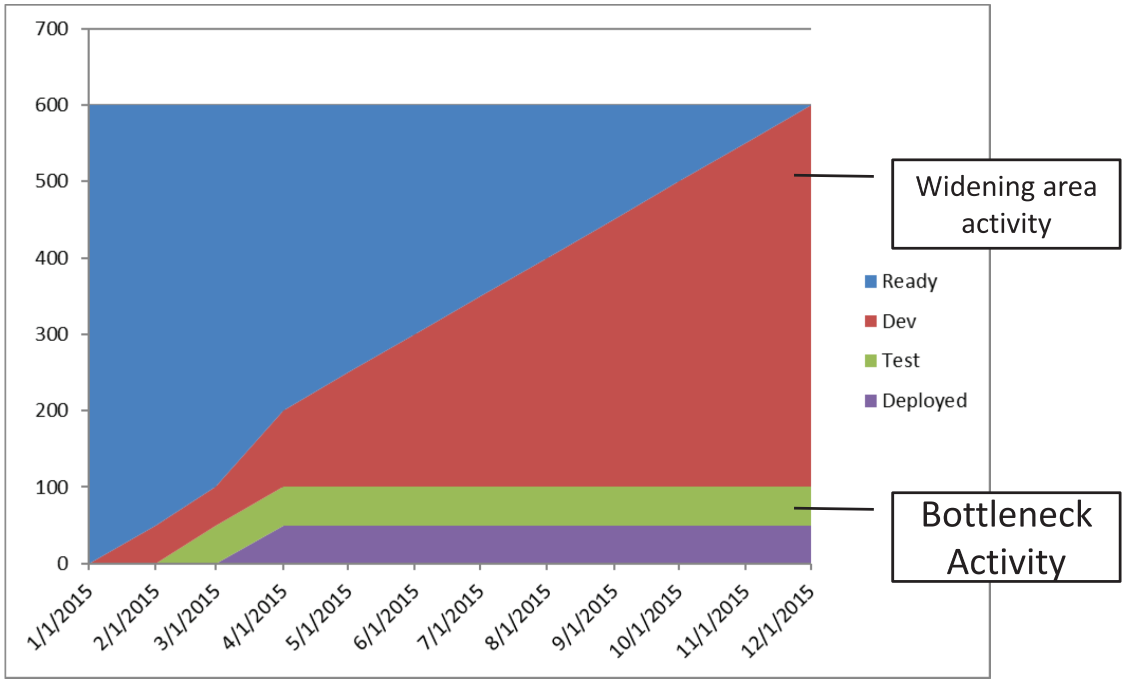
Cumulative Flow Diagrams (CFD’s)

Bottlenecks and Theory of Constraints



Cumulative Flow Diagrams (CFD's)

Bottlenecks and Theory of Constraints



Agile Contracting

Agile’ s flexibility creates difficulty in outlining contract acceptance criteria

- Agile attempts to fix resources and time (cost) and vary functionality

“Customer collaboration over contract negotiation”

- Close cooperation
- Active participation
- Timely and often feedback

Money for nothing and change for free

Agile Contracting

Graduated Fixed Price Contract

- Buyer / Seller share in risks and rewards
- Different hourly rates based on:
  - Finish early, Finish on time, Finish late

Fixed Price Work Packages

- Mitigate risks of under/over estimating

Verifying and Validating Value

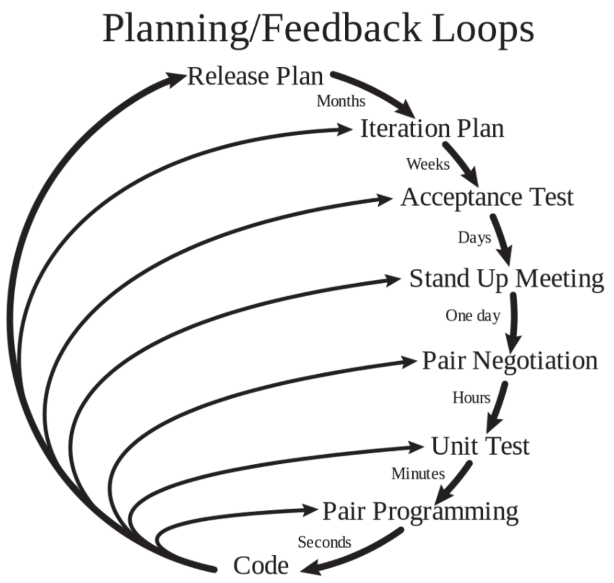
"Gulf of Evaluation"

- What one person describes is often different from how another interprets

Frequent Verification and Validation

Resolve problems as soon as possible

Don’t let little problems grow over time



Stakeholder Engagement

DOMAIN 3

Stakeholder Stewardship

Looking after everyone involved on the project

Ensuring everyone has everything they need to compete the project successfully

Starts with identifying the stakeholders

Educating People about Agile

Teach all the stakeholders about the benefits of agile

Concerns about agile can Include:

- Senior management and sponsor: They are worried about the risk of failing
- Managers: fear the loss of control
- Project team: resist agile methods
- Users: will not get all features

Engaging Stakeholders

Short iterations and release keeps them engage

Keeping then engage can lead to stakeholders being more involved and getting more change request

This helps us to identify risk and issues early

If some stakeholders are causing problems, the agile PM will need to use their interpersonal skills to resolve issues

Need to have a process for escalating stakeholders issues

Why such a big focus on stakeholders?

- Projects and done by people for people

Methods of Stakeholder Engagement

Get the right stakeholders

Cement stakeholder involvement

Actively manage stakeholder interest

Frequently discuss what done looks like

Show progress and capabilities

Candidly discuss estimates and projections

Set a Shared Vision

Important to ensure customers and agile project team has the same vision

Methods include:

- Agile Charter
- Definition of “Done”
- Agile Modeling
  - Use case diagram
  - Data models
  - Screen design
- Wireframes
- Personas

Agile Chartering

High-level (uses the W5H)

Agreement

Authority to proceed

Focuses on *how* project will be conducted

- Allows for flexibility and ability to deal with change

Project specific processes outlined

May use project Tweet – Describes project goal in 140 Characters or less.

Definition of "Done"

Creating a shared vision of what done looks like

Should be done for:

- User stories
- Releases
- Final project deliverables